Computing in lattice ordered groups and related structures

Peter Jipsen

Chapman University, Orange, California, USA

June 16, 2009

Outline

- Groups and ℓ-groups on computers
- Free abelian ℓ-groups on computers
- Proofs about ℓ-groups on computers
- Holland's embedding theorem
- The Holland-McCleary algorithm in Python
- Generalizations of ℓ-groups

Most computer languages provide native implementations of a subset of the integers with $+, -, \min, \max$

E.g. c, c++, Java, Fortran have an integer type that ranges from -2^{31} to $2^{31} - 1$ (-2147483648 to 2147483647)

What happens when the result of a $+$ is larger than this value?

c rolls over to give a negative result (calculates in $\mathbb{Z}_{2^{32}}$)

Most other languages give overflow error

Computers also provide *approximations* of real numbers.

Systems for computing with groups: GAP, Sage, Magma, (Maple, Mathematica)

GAP is open source, and is part of the free computer algebra system Sage

GAP provides finite permutation groups, matrix groups, free groups, some support for finitely presented groups

Also works with many other mathematical structures (rings, character tables, fields, polynomials, monoids, ...)

Does not provide ℓ-groups, except integers, rationals (limited to fit in memory)

Another useful tool for computing in *finite* algebras is the Universal Algebra Calculator (R. Freese, E. Kiss, M. Valeriote)

Sage is a free computer algebra system, used via a web browser

Based on the programming language Python – very mathematical

Size of integers only limited by available memory

Accuracy of approximations of reals only limited by available memory

The philosophy of Sage is "use the best available open-source mathematical packages and make them work together well; don't reinvent the wheel!"

E.g. Sage includes GAP, Maxima, Singular, R, numpy,...

It is being developed by math professors and graduate students around the world

Very easy to install, if you are interested go to sagemath.org

$\mathbb{R}$ is an abelian ℓ-group that generates the variety $\mathcal{A}$ of all ℓ-groups

By Birkhoff, the free algebra in $\mathcal{A}$ with set of generators X can be realized in the algebra of functions $\mathbb{R}^X \to \mathbb{R}$

Each $x \in X$ is mapped to the projection π_x given by $\pi_x(f) = f(x)$

E.g. If $X = \{x, y\}$ then $F_{\mathcal{A}}(x, y)$ is embedded in $\mathbb{R}^2 \to \mathbb{R}$ with x, y mapping to the projections π_1, π_2

So calculations in finitely generated free abelian ℓ-groups are done with piecewise linear functions

Let's take a look.

The decision procedure for abelian ℓ-groups is also based on linear algebra

Want to decide if $s(\mathbf{x}) = t(\mathbf{x})$ holds in $\mathcal{A}$ where $\mathbf{x} = x_1, \ldots, x_n$

Equivalently, decide if $w(\mathbf{x}) = 0$ holds in $\mathcal{A}$ for an ℓ-group term w

i.e. for all $\mathbf{x}$ $\quad w(\mathbf{x}) \leq 0$ and $-w(\mathbf{x}) \leq 0$

So check if $\exists \mathbf{x} \quad (\mathbf{x}) > 0$, write $w(\mathbf{x}) = \bigvee_k \bigwedge_i \sum_j a_{kij} x_j$

Equivalently: $\exists k \; \exists \mathbf{x} \;\; A_k \mathbf{x} \geq \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$ where $A_k = [a_{kij}]$

This means we want to solve linear inequalities with integer coefficients

Integer linear programming is an approach for finding (optimal) solutions of

$A\mathbf{x} \geq \mathbf{b}$ for $A \in \mathbb{Z}^{m \times n}$, $\mathbf{b} \in \mathbb{Z}^m$ and $\mathbf{x} \in \mathbb{Z}^n$

There are lots of techniques and many software packages that implement (integer) linear programming

With real-valued coefficients e.g. the simplex method works well, and other methods are even polynomial time

But integer linear programming is NP-complete in general

A Sage package lp_solve is available (soon part of Sage)

Let's see how it works

Computing in non-abelian ℓ-groups requires additional effort to implement

Can work with finitely generated subgroups of order-preserving automorphisms of $\mathbb{R}$

This is similar to what GAP provides for finite permutation groups

Thanks to Charles Holland's fundamental result that every ℓ-group embeds in the order-automorphisms of a chain, every finitely generated ℓ-group is obtained this way

For example, take $f(x) = 2x$ and $g(x) = x^3$, what can we say about the ℓ-group generated by these order automorphisms?

E.g. is it normal valued?

A Python implementation of the Holland-McCleary algorithm for deciding ℓ-group equations

Holland and McCleary, *Solvability of the word problem in free lattice-ordered groups*, Houston Journal of mathematics, **5**(1), (1979) p. 99–105.

Rather than expanding ℓ-group terms to standard form (joins of meets of group terms), the algorithm operates directly on the more compact "*inverse normal form*" (inverses occur only on variables).

It suffices to consider equations of the form $t = 1$, where t is in inverse normal form. It is also convenient to allow inequations of the form $t \geq 1$.

According to Holland and McClearly, the decision procedure amounts to evaluating t in the ℓ-group of *order-automorphisms of a dense linear order*, since this ℓ-group generates the variety of all ℓ-groups.

The *main routine* is of the form **eval**(t, d, i)

where t is the ℓ-group term, d is a *list of group words* $(d(0), \ldots, d(m-1))$ in *(group) normal form*, and i is an index to one of the members of the list.

The pair (d, i) is called an *(arrow) diagram* and represents a set of evaluations of the ℓ-group term.

Concretely, it codes up all *assignments* h from variables to order-automorphisms of the dense linear order that satisfy

$$h(d(0))(0) < h(d(1))(0) < \cdots < h(d(m-1))(0)$$

This requires that a diagram (d, i) satisfies some constraints:

(a) The group identity 1 is an element in d.

(b) If w is a group word in d, then each initial segment of w is also a group word in d.

(c) All group words are in normal form, and there are no duplicates in d.

(d) For any variable x, if $d(i)x = d(j)$ and $d(k)x = d(l)$ then $i < k$ iff $j < l$.

Constraint (d) ensures that the diagram represents an assignment to possible **order**-automorphisms.

E.g. check the equation for normal valued ℓ-groups does not hold in all ℓ-groups

$x \geq 1$ and $y \geq 1 \implies yx \leq x^2y^2$

Construct the following diagrams: $((\underline{1}), 0)$, $((1, \underline{x}), 1)$

$((1, x, \underline{x^2}), 2)$, $((1, x, x^2, \underline{x^2y}), 3)$, $((1, x, x^2, x^2y, \underline{x^2y^2}), 4)$,

$((1, x, x^2, \underline{x^2y^2x^{-1}}, x^2y, x^2y^2), 3)$,

$((\underline{x^2y^2x^{-1}y^{-1}}, 1, x, x^2, x^2y^2x^{-1}, x^2y, x^2y^2), 0)$

The output of the eval(t, d, i) routine is a set of pairs:

$\{(d_1, i_1), \ldots, (d_n, i_n)\}$

that represents all possible evaluations of t, starting from, and extending, (d, i).

This routine satisfies the following recursive equations:

$$\text{eval}(1, d, i) = \{(d, i)\}$$

$$\text{eval}(x, d, i) = \begin{cases} \{(d, j)\} & \text{if } d(i)x = d(j) \\ \{(c_1, i_1), \ldots, (c_n, i_n)\} & \text{otherwise} \end{cases}$$

where each (c_k, i_k) is a diagram obtained from (d, i) by inserting the group normal form of $d(i)x$ in position i_k, and the list represents all possible diagrams that can be obtained from (d, i).

$$\text{eval}(st, d, i) = \bigcup\{\text{eval}(t, c, j) : (c, j) \in \text{eval}(s, d, i)\}$$

$$\text{eval}(s \vee t, d, i) = \{(c, \max(l, m)) : (c, l) \in \text{eval}(t, b, k) \text{ for some}$$

$$(b, j) \in \text{eval}(s, d, i) \text{ and } b(k) = d(i) \text{ and } c(m) = b(j)\}$$

$$\text{eval}(s \wedge t, d, i) = \{(c, \min(l, m)) : (c, l) \in \text{eval}(t, b, k) \text{ for some}$$

$$(b, j) \in \text{eval}(s, d, i) \text{ and } b(k) = d(i) \text{ and } c(m) = b(j)\}$$

And this is how it looks in Python:

```
def newword(w,v):
.       if len(w)>0 and w[-1]==inverse[v]: return w[:-1]
.       else: return w+v
def evaluate(t,d,i): # evaluate l-group term t in Aut(Q)
.       if not hasattr(t,'s'): return [[d,i]]
.       if t.a==(): # t is a variable
.               v = t.s
.               w = newword(d[i],v)
.               if w in d: return [[d,d.index(w)]]
.               j = i+1; n = len(d)
.               while j<n and newword(d[j],v) not in d: j = j+1
.               if j<n: n = d.index(newword(d[j],v))
.               j = i-1; k = 0
.               while j>=0 and newword(d[j],v) not in d: j = j-1
.               if j>=0: k = d.index(newword(d[j],v))+1
.               D = []
.               for j in range(k,n+1): D = D + [[d[:j]+[w]+d[j:],j]]
.               return D
```

```
    if t.s=="*" or t.s=="+":
        E = evaluate(t.a[0],d,i)
        F = []
        for e in E:
            F = F + evaluate(t.a[1],e[0],e[1])
        return F
    if t.s=="|" or t.s=="&":
        E = evaluate(t.a[0],d,i)
        G = []
        for e in E:
            j = e[0].index(d[i])
            F = evaluate(t.a[1],e[0],j)
            for f in F:
                k = f[0].index(e[0][e[1]])
                if t.s=="|": G = G + [[f[0],max(k,f[1])]]
                else: G = G + [[f[0],min(k,f[1])]]
        return G
```

The final decision for $t = 1$ or $t \geq 1$ is reached by examining the set of diagrams produced by $\mathrm{eval}(t, (1), 0)$.

If there exists a diagram (d, i) in this set such that $d(i) \neq 1$, then $t = 1$ fails, and

if there exists a diagram (d, i) such that $d(k) = 1$ and $k > i$, then $t \geq 1$ fails.

The current implementation has not yet been optimized.

If the equation fails, the *failure can be checked by an independent program* (or directly by hand).

If the equation holds, a *theorem prover could be used to find a (quasi-equational) proof* .

Let's see how it works.

The decision procedure for ℓ-group equations can easily be extended to cover the variety of negative cones of ℓ-groups, integral GMV-algebras, and GMV algebras (Galatos, Tsinakis 2005)

The same translation can be used to adapt the decision procedure for abelian ℓ-groups to decide MV equations, but can also use linear programming directly

Further results on decidability

C. Holland, S. H. McCleary 1979: ℓ-groups have decidable equation theory

A.M.W. Glass, Y. Gurevich 1983: ℓ-groups have undecidable word problem

N. G. Hisamiev 1966: abelian ℓ-groups have decidable universal theory, by V. Weispfenning 1986, in fact co-NP-complete

but by Y. Gurevich 1967, the first-order theory is hereditarily undecidable

MV-algebras have the finite embeddability property (FEP), so the univeral theory is decidable

IGMV and GMV have decidable equational theory because of a connections to ℓ-groups (N. Galatos, C. Tsinakis 2005), but no finite model property (FMP)

P. Jipsen and F. Montagna 2006, 2008: GBL and IGBL do not have FMP but normal GBL-algebras have FEP

Further results on decidability

C. Holland, S. H. McCleary 1979: ℓ-groups have decidable equation theory

A.M.W. Glass, Y. Gurevich 1983: ℓ-groups have undecidable word problem

N. G. Hisamiev 1966: abelian ℓ-groups have decidable universal theory, by V. Weispfenning 1986, in fact co-NP-complete

but by Y. Gurevich 1967, the first-order theory is hereditarily undecidable

MV-algebras have the finite embeddability property (FEP), so the univeral theory is decidable

IGMV and GMV have decidable equational theory because of a connections to ℓ-groups (N. Galatos, C. Tsinakis 2005), but no finite model property (FMP)

P. Jipsen and F. Montagna 2006, 2008: GBL and IGBL do not have FMP but normal GBL-algebras have FEP

Further results on decidability

C. Holland, S. H. McCleary 1979: ℓ-groups have decidable equation theory

A.M.W. Glass, Y. Gurevich 1983: ℓ-groups have undecidable word problem

N. G. Hisamiev 1966: abelian ℓ-groups have decidable universal theory, by V. Weispfenning 1986, in fact co-NP-complete

but by Y. Gurevich 1967, the first-order theory is hereditarily undecidable

MV-algebras have the finite embeddability property (FEP), so the univeral theory is decidable

IGMV and GMV have decidable equational theory because of a connections to ℓ-groups (N. Galatos, C. Tsinakis 2005), but no finite model property (FMP)

P. Jipsen and F. Montagna 2006, 2008: GBL and IGBL do not have FMP but normal GBL-algebras have FEP

Further directions of research

An embedding of ℓ-groups into distributive residuated lattices of binary relations with composition as multiplication.

Suppose $h : G \to Aut(\Omega)$ is an embedding. Define

$r : G \to \mathcal{P}(\Omega^2)$ by $r(g) = \{(u, v) : f(u) \leq v\}$

Then it is easy to check that this is a residuated lattice embedding.

This construction generalizes order-automorphisms of a chain, and provides new examples of involutive residuated lattices that are "close" to ℓ-groups but are non-cyclic.

These algebras are called ℓ-pregroups by Lambek (1999) and have been studied in several papers by Buszkowski (2001-2008)

A *pregroup* is a structure $(G, \cdot, 1, \sim, -)$ such that $(G, \cdot, 1, \leq)$ is a partially ordered monoid and

$$-x \cdot x \leq 1 \leq x \cdot -x \qquad \text{and} \qquad x \cdot \sim x \leq 1 \leq \sim x \cdot x$$

Note that if $-x = \sim x$ then the pregroup is a po-group since then $-x \cdot x = 1 = x \cdot -x$

A pregroup is called *proper* if it is not a po-group

Lemma: Any proper pregroup is unbounded, nonabelian and not totally ordered

An *ℓ-pregroup* is a pregroup in which partial order is a lattice order

As for ℓ-groups, it is easy to see that ℓ-pregroups are residuated lattices

Theorem: ℓ-pregroups are termequivalent to involutive residuated lattices that satisfy the identity $x \cdot y = x + y$ where $x + y = \sim(-y \cdot -x)$

Problem: Are the lattice reducts of ℓ-pregroups distributive?

Problem: Investigate the structure of free ℓ-pregroups. Even the 1-generated case is not well understood

Eventually ℓ-groups are going to be a standard part of research software like GAP and Sage

This will be in large part because of the fundamental results of Charles Holland

After that, we will concentrate on making computers understand the *significance* of Charles's results on the theory of ordered algebras

As for ℓ-groups, it is easy to see that ℓ-pregroups are residuated lattices

Theorem: ℓ-pregroups are termequivalent to involutive residuated lattices that satisfy the identity $x \cdot y = x + y$ where $x + y = \sim(-y \cdot -x)$

Problem: Are the lattice reducts of ℓ-pregroups distributive?

Problem: Investigate the structure of free ℓ-pregroups. Even the 1-generated case is not well understood

Eventually ℓ-groups are going to be a standard part of research software like GAP and Sage

This will be in large part because of the fundamental results of Charles Holland

After that, we will concentrate on making computers understand the *significance* of Charles's results on the theory of ordered algebras

As for ℓ-groups, it is easy to see that ℓ-pregroups are residuated lattices

Theorem: ℓ-pregroups are termequivalent to involutive residuated lattices that satisfy the identity $x \cdot y = x + y$ where $x + y = \sim(-y \cdot -x)$

Problem: Are the lattice reducts of ℓ-pregroups distributive?

Problem: Investigate the structure of free ℓ-pregroups. Even the 1-generated case is not well understood

Eventually ℓ-groups are going to be a standard part of research software like GAP and Sage

This will be in large part because of the fundamental results of Charles Holland

After that, we will concentrate on making computers understand the *significance* of Charles's results on the theory of ordered algebras

As for ℓ-groups, it is easy to see that ℓ-pregroups are residuated lattices

Theorem: ℓ-pregroups are termequivalent to involutive residuated lattices that satisfy the identity $x \cdot y = x + y$ where $x + y = \sim(-y \cdot -x)$

Problem: Are the lattice reducts of ℓ-pregroups distributive?

Problem: Investigate the structure of free ℓ-pregroups. Even the 1-generated case is not well understood

Eventually ℓ-groups are going to be a standard part of research software like GAP and Sage

This will be in large part because of the fundamental results of Charles Holland

After that, we will concentrate on making computers understand the *significance* of Charles's results on the theory of ordered algebras

References I

W. Buszkowski, *Pregroups: Models and grammars*, in Relational Methods in Computer Science, LNCS 2561, (2002), 35–49

N. Galatos, P. Jipsen, T. Kowalski and H. Ono, Residuated Lattices: an algebraic glimpse at substructural logics, Studies in Logics and the Foundations of Mathematics, Elsevier, 2007

N. Galatos and C. Tsinakis, *Generalized MV-algebras*, Journal of Algebra **283** (2005), 254–291.

C. Holland, *The lattice-ordered group of automorphisms of an ordered set*, Michigan Math. J. 10 (1963), 399–408

C. Holland and S. McCleary, *Solvability of the word problem in free lattice-ordered groups*, Houston Journal of mathematics, **5**(1), (1979) p. 99–105.

P. Jipsen and C. Tsinakis. *A survey of residuated lattices*, in "Ordered Algebraic Structures" (J. Martinez, editor), Kluwer Academic Publishers, Dordrecht, 2002, 19–56

P. Jipsen and F. Montagna, *On the structure of generalized BL-algebras*, Algebra Universalis, 55 (2006), 226–237

P. Jipsen and F. Montagna, *The Blok-Ferreirim theorem for normal GBL-algebras and its application*, Algebra Universalis, 60, to appear